INTERACTIVE LEARNING MODULES FOR 
FEEDBACK FUNDAMENTALS USING PID 
CONTROL

J.L. Guzmán, K. J. Åström, S. Dormido, T. Hägglund, M. Berenguel, and Y. Piguet

INTRODUCTION

Looking back thirty years ago, our society has been immersed in a continuous technological advance. Specially, the great advances of the New Information and Communication Technologies (NICT) in last years have been reflected in the society in a general context. Multiple effects can be observed at business level (e.g., remote control, remote management, flexible timetable), socio-cultural level (e.g., mobile telephone, electronic bank, digital television), and at teaching level (e.g., digital sliders, distance education, interactive information, virtual and remote labs) opening an innumerable number of possibilities. On the other hand, the mode in which the information is shown to the users is changing, where data and images are not only presented as static elements, but also as interconnected elements with some specific functionalities. This feature is known as Interactivity which allows enhancing the users’ motivation through a more
participating activity. Although Interactivity can be defined in several ways, in the sense of Interactive Tools, it can be defined as a collection of graphical windows whose components are active, dynamic, and/or clickable; and that is intended to explain just a few concepts. The use of interactive and instructional graphic tools would reinforce active participation of user [4], [12].

The idea of changing properties and immediately being able to see the effects is very powerful both for learning and for designing in the Automatic Control field [2], [9]. In spite of all the advances in control theory, the PID controller is still the workhorse of control which can be used to solve a large variety of control problems. So, this work describes three interactive learning modules which have been developed to permit obtaining a good intuition and working knowledge of PID control. The design of the modules was developed considering all common aspects with the aim of performing similar structured interfaces. The modules consist of menus where process transfer functions and PID controllers can be chosen, parameters can be set, and results stored and loaded. A graphic display which shows time or frequency responses is a central part. The graphics can be manipulated directly by dragging points, lines, and curves or by using sliders. Parameters that characterize robustness and performance are also displayed. All modules have two icons to access Instructions and Theory. Instructions give access to a document which contains suggestions for exercises, and Theory provides access to relevant theory via Internet. The modules can be used in many different ways, one extreme is a full-fledged exercise with serious analysis and reporting, another is simply free experimentation following own ideas. The modules can be used in lectures and, as will be shown in the next sections, multiple exercises can be proposed to give a good intuitive understanding of the properties of PID Control.
These modules can be viewed as an attempt to make the key pictures in the book *Advanced PID Control* [1] interactive. The idea was to develop interactive learning tools which could be used for introductory control courses at universities and other schools, and for engineers in industry. The modules should be self-contained, suitable both for self-study, courses, and for demonstrations in lectures, and they should not require any additional software.

The modules are implemented in *Sysquake* [11], a *Matlab*-like language with fast execution and excellent facilities for interactive graphics. The modules are free and available on an own web site [3] and also on the web of Sysquake environment [11] (section Applications) for Windows, Mac, and Linux operating systems.

The implementation of the modules is reasonably straight forward. Manipulation of graphical objects are well supported in Sysquake. Numerics for simulation consist of solving linear differential equations with constant coefficients and simple nonlinearities representing the saturations. For linear systems the complete system is sampled at constant sampling rate and the sampled equations are iterated. For systems with saturation the process and the controller are sampled separately with first order holds, the nonlinearities are added, and the difference equations are then iterated. Delays are simply dealt with because the sampled systems are difference equations of finite order.

Next sections describe the three developed modules: the central module is called *PID Basics*, two auxiliary modules *PID Loop Shaping* and *PID Windup* illustrate loop shaping and windup. One consideration that must be kept in mind is that the main feature of the tools (Interactivity) cannot be easily illustrated in a written text. Nevertheless, some of the advantages
Figure 1. The user interface of the module *PID Basics*. The plots show the time response of the Gang of Six.

of the applications are shown. The authors cordially invite to visit the web site to experience the interactive features of the tools [11].

**PID Basics**

A simple and intuitive way to understand PID control is to look at the responses of the closed-loop system in the time domain and to observe how the responses depend on the
controller parameters. In order to have a reasonably complete understanding of a feedback loop, it is essential to consider six responses, the Gang of Six [1]. One possibility is to show process output and controller output for step commands in set-point and load disturbances, and the response to noise in the sensor. The study of feedback in this way can be performed using PID Basics. Frequency responses and mixed time and frequency results can also be shown.

The interaction is straightforward because it is done mainly by using sliders for controller parameters. Process models can be chosen from a menu which contains a wide range of transfer functions. It is also possible to enter an arbitrary transfer function in the Matlab rational function format. Process gain and time delay can be changed interactively. The PID controller has the structure represented by

\[ U(s) = K \left( bY_{sp} - Y + \frac{1}{sT_i} E - \frac{sT_d}{1+sT_d/R} Y \right), \]  \hspace{1cm} (1)

where \( K \) is the proportional gain, \( T_i \) the integral time, \( T_d \) the derivative time, \( R \) the parameter of derivative term filter, \( b \) is a set-point weight, and \( U, Y_{sp}, Y \) and \( E \) are the Laplace transforms of control signal \( u \), set-point \( Y_{sp} \), process output \( y \) and control error \( e = Y_{sp} - y \), respectively.

*Description of the Interactive Tool*

This section briefly describes the main aspects of PID Basics. The main screen of the tool is shown in Figure 1.

*Process:* It corresponds to the parameter group located on the left hand side of the tool screen, just below the icons (see Figure 1). It contains the information about the process to control,
showing a symbolic representation of the transfer function and several interactive elements to change the process parameters. From Figure 1 it can be seen that the current example is a fourth order process with the transfer function

\[ G(s) = \frac{K_p}{(s+1)^n} \]

where the interactive parameters are given by \( K_p \) (by means of a slider) and \( n \) (by means of a text edit). When the user modifies any plant parameter, the symbolic representation is immediately updated, being its effect reflected on the rest of IT elements. The user can modify the transfer function of the process from the Settings menu as will be shown later.

**Controller:** Five radio buttons are available to select the desired controller. The options to choose are proportional (P), integral (I), proportional-integral (PI), proportional-derivative (PD), and proportional-integral-derivative (PID). Several sliders are available below the radio buttons in order to modify the controller parameters. The number of sliders shown depends on the chosen controller. For example, Figure 1 shows five sliders since the option selected is a PID controller \((K, T_i, T_d, \xi, \text{ and } b)\).

**Performance and robustness information:** Some measures about performance and robustness are provided in order to study the control designs. The performance category is classified into three groups: set-point response, load disturbances, and noise response. For the set-point response the integral absolute error (IAE) and overshoot (overshoot) measures are given. Integral absolute error (IAE), integral gain (ki), maximum error (emax), and the time to reach the maximum \((t_{\text{max}})\)
are the information provided for load disturbances. The integral absolute errors and the maximum error values are normalized to unit step changes in setpoint and load disturbances. The response to noise is characterized by the standard deviations of the signals \( x \) (signal without noise, \( \text{sigma}_x \)), \( y \) (signal with noise, \( \text{sigma}_y \)), and \( u \) (control signal, \( \text{sigma}_u \)). The robustness measures are maximum sensitivity (\( \text{Ms} \)), maximum complementary sensitivity (\( \text{Mt} \)), gain margin (\( \text{Gm} \)), and phase margin (\( \text{Pm} \)). This information can be duplicated in order to compare two designs as will be shown later. A deeper description of these measures can be found in reference [1].

**Graphics:** A couple of graphics are shown on the right hand side of the tool (see Figure 1). These graphics have three representation modes depending on the selected option from Settings menu. These modes are *time domain*, *frequency domain*, and *frequency/time domain*.

The *time domain* mode is that shown in Figure 1, where the time responses for the system output (*Process Output*) and input (*Controller Output*) are displayed, so providing all the information evolved in the *Gang of Six* [1]. There are several interactive graphical elements on the graphics to interact with the application. The vertical green line located at time \( t = 0 \) allows modifying the set-point amplitude. The green and black vertical lines located in the middle of the graphics allow setting the value and instant time for load disturbances and measurement noise respectively. The vertical and horizontal scales can be modified using three black triangles available on the graphics (\( \bullet \), \( \bigtriangledown \)). For example, in Figure 1 the set-point is set to 1, the load disturbance to 0.9 at instant time 32, and the measurement noise to 0.02 at instant time 60. It is also possible to know the value for the input or output signal at an specific instant time, being only necessary to place the mouse over the curve. Figure 1 shows an example where for the
instant time $t = 37.78$, the output and input signals are 1.62 and 0.38 respectively. Notice that all the previous options are available from both graphics, Process Output and Controller Output.

On the top of the Process Output graphic, there are two checkboxes called save and delete. These buttons make it easy to store a simulation for comparison. If the save button is selected, the current design is saved and kept on the graphics in blue color. Then, a new design in red color appears allowing to perform a comparison between both designs. Performance and robustness parameters are duplicated showing the values in red and blue colors associated to each design. Legends at Process Output and Controller Output graphics are also shown with the value of controller parameters for both designs. Figure 1 shows an example which compares response of PI ($K = 0.43, T_i = 2.27, b = 0$) and PID ($K = 1.13, T_i = 3.36, T_d = 1.21, b = 0.54, \kappa = 10$) controllers for a process with the transfer function $P(s) = 1/(s+1)^4$. The PID controller gives a better response to load disturbances by reacting faster, but the noise also generates more control action. The delete option can be selected in order to remove the no-desired design. Note that if the transfer function of the process or some input signal (set-point, load disturbance, and measurement noise) are modified, both designs are interactively affected. The number of saved designs has been constrained to two in order to avoid complicated comparisons.

The last options for the time domain mode are shown on the top of Controller Output graphic. Such options show the proportional (P), integral (I), and derivative (D) signals of the controller. Figure 2 shows an example where the black signal represents the proportional action, the blue one the integral action, the pink one the derivative action, and the red one the PID control signal.
Figure 2. PID actions on graphic Controller Output.

(a) Frequency Domain  
(b) Frequency and Time domains simultaneously

Figure 3. Time and Frequency domains on the Interactive Tool.

The aspect of the frequency domain mode is shown in Figure 3. Once this mode is selected from the Settings menu, the left side of the tool remains untouched, only changing the right side where the time response graphics are replaced by frequency domain ones, Transfer function Magnitude and Transfer function Phase. The graphics allow interactively modifying
the vertical and horizontal scales in the same way that in the time domain, and also visualizing the magnitude and phase for a specific frequency $\omega$ placing the mouse over the signals. Figure 3(a) shows an example where $|S_n(\omega)| = 3.46$ and $\angle S_n(\omega) = 51.73$ (the value of $\omega = 2.15$ rad/s is shown in the status bar at the bottom of the tool). The frequency response for the Gang of Six transfer functions [1] plus the open loop transfer function $L(i\omega) = P(i\omega)C(i\omega)$ can be shown on the graphics using checkboxes placed on the top of Transfer function Magnitude graphic (the names of the checkboxes are shown using the relation of the sensitivity and complimentary sensitivity functions with the rest of transfer functions. e.g. $PS = P\frac{1}{1+PC}$). Figure 3(a) shows an example where all transfer functions are shown.

It is also possible to show time and frequency domains simultaneously. An example can be seen in Figure 3(b). The upper graphic represents time domain, and the lower one represents frequency domain. The default screen shows the output and the magnitude for the time and frequency domains, respectively. However, on the top of the graphics, there exists a couple of radio buttons which permit choosing between the output or input (for time domain), and magnitude or phase (for frequency domain). This mode is very interesting since it is possible to observe the effect of parameter modifications on both domains simultaneously.

Settings menu: The Settings menu is available on the top menu of PID Basics, and it is divided into six groups. From the first entry, Process transfer function, several processes can be selected, being also available an option to enter an arbitrary transfer function in the numerator (num) and denominator (den) form used in Matlab. Specific values for controller parameters can be entered using the Controller parameters menu option. The third entry, Time/Frequency
domain, allows choosing between the three modes commented above: Time domain, Frequency domain, and Both domains. Results can be stored and recalled using the Load/Save menu (using the options Save design and Load design respectively). The option Save report helps to save all essential data in ascii format, this possibility being useful for documenting results. The menu selection Simulation fits the simulation time, the maximum time delay (in order to avoid slow simulations), and to active the Sweep option to show the results for several controller parameters simultaneously; that is, it is possible to study the effect of any controller parameter between specific minimum and maximum values. This last option is only available in time domain mode. When it is active, new radio buttons appear in the controller parameters zone to permit the selection of the desired parameter to sweep. The last menu option, Examples Advanced PID Book, allows loading examples from the book [1], in such a way that the user can begin with such examples to explore what happens when the parameters are modified.

Illustrative Examples

Some examples extracted from reference [1] are presented in order to test the capabilities of the tool.

Set-point response: The response to set-points is important when making grade changes in process control. Tracking set-point is a key issue in motion control. The purpose of this example is to explore how the set-point response of the system is influenced by the controller parameters. The load disturbance and noise amplitude are set to zero using the interactive vertical lines. The transfer function of the process is given by
\[ G(s) = \frac{1}{(0.12s + 1)(0.25s + 1)(0.5s + 1)(s + 1)} \]

Figure 4(a) shows the result of controlling the process using a P-controller for different proportional gains (using the **Sweep** option menu). So, it is possible to observe how the output doesn’t reach the set-point, and the steady-state error depends on the controller gain in the form \( G(0) = (KK_p)/(1 + KK_p) \).

![Graphs showing set-point response](image)

(a) Proportional controller  
(b) Integral controller

Figure 4. Example of set-point response. Proportional (left) and Integral controller (right).

An I-controller can be tested for different values of the integral gain as shown in Figure 4(b). In this case the set-point is reached but the closed-loop response is too sluggish. If a PI-controller is used the response is going to perform better getting characteristics from previous controller, fast response from P-controller and free steady-state error from I-controller. Figure 5 shows a comparison for a PI-controller (\( K = 0.38, T_i = 1.34, b = 1 \)) with the P (\( K = 0.38, b = 1 \))
and I ($T_i = 2.43$) controllers. It can be seen how the system reaches faster the set-point obtaining better integral absolute error, $IAE = 1.61$ instead of $IAE = \infty$ for P-controller, and $IAE = 3.79$ for I-controller. Finally, notice that the set-point response can be done slower and faster playing with the tracking parameter $b$, as Figure 6 shows.

**Load Disturbance Response:** Load disturbances are typically low frequency signals that drive the system away from its desired behavior. The response to load disturbances is a key issue in process control, since most controllers attempt to keep process variables close to desired set-
points [1]. The purpose of the following example is to investigate the effects of load disturbances and how their effect is influenced by the controller type and parameter settings. The set-point and noise amplitudes have been set to zero, and the amplitude of the load disturbance has been set to 0.9 at instant time $t = 0$. The process transfer function is given by

$$G(s) = \frac{1}{(s + 1)^4}$$

Firstly, a comparison between a P-controller ($K = 0.6, b = 1$) and PI-controller ($K = 0.6, T_i = 2, b = 1$) is displayed in Figure 7. It can be observed how the P-controller doesn’t eliminate the effect of the disturbances, while the PI-controller does. This fact can be corroborated from the transfer function $G_{yd} = \frac{Y}{D} = \frac{P}{1 + PC}$, where for a P-controller that $G_{yd}(0) = \frac{K}{1 + K_{pp}} \simeq 0$, and for a PI-controller $G_{yd}(0) = 0$.

![Figure 7. Example of load disturbance response. P and PI controllers.](image)

So, as commented above, the response of the process variable to load disturbances is given by [1]
\[ G_{yd} = \frac{P}{1+PC} = PS = \frac{T}{C} \]

For a system with \( P(0) \neq 0 \) and a controller with integral action the previous transfer function can be approximated by [1]:

\[ G_{yd} = \frac{T}{C} \approx \frac{1}{C} \approx \frac{C}{k_i} \]

where \( k_i = \frac{K}{l_i} \) is the integral gain. Since load disturbances typically have low frequencies the previous equation is a good measure of load disturbance rejection. So, large values of \( k_i \) will provide adequate load disturbance responses.

Figure 8 shows the load disturbance responses for two PI controllers with \( k_i \) values of 0.36 (in red color) and 0.30 (in blue color). As the previous equation indicates, the controller with larger integral gain leads to better results to load disturbances, obtaining a faster response and smaller values for IAE and emax, but the stability margins are reduced, as can be observed from the robustness parameters. Figure 9 shows the frequency responses of \( G_{yd} \) and \( S \) for two PI controllers with large (\( k_i = 0.85 \)) and small (\( k_i = 0.30 \)) values of \( k_i \). From this figure, it can be noticed that large values of \( k_i \) imply large peaks of the sensitivity function. So, it is necessary to reach a trade-off between load disturbance rejection and robustness.

Some tuning methods allow obtaining good compromises between robustness and load disturbance response. AMIGO (Approximate M constrained Integral Gain Optimization) [1], [5].
Figure 8. Example of load disturbance response. Integral gain \((k_i)\) influence.

Figure 9. Example of load disturbance response. Frequency domain responses of \(G_{3vl}\) and \(S\) for a PI with \(k_i = 0.85\) (left) and \(k_i = 0.30\) (right).

[6], [7] is one of these methods that, in the same way as the well-known Ziegler and Nichols method [13], [14], focusses on load disturbances by maximizing integral gain but also adding a robustness constraint. For the previous example, the result of applying this method is shown in Figure 10. The AMIGO-step method has been used to design a PI controller with \(K = 0.414\) and \(T_i = 2.66\). A slow response to load disturbances is obtained but with good stability margins.

Response to measurement noise: Measurement noise is a disturbance that distorts the information about the process obtained by the sensors. Measurement noise typically has high
Figure 10. Example of load disturbance response. PI controller using AMIGO-step method.

Figure 11. Example of measurement noise response. PID controllers with $\kappa = 1.5$ and $\kappa = 10$.

frequencies, and is fed into the system by feedback. This phenomenon creates control actions and variations in the process output, being important that measurement noise does not generate too large control actions. The next example shows how measurement noise affects the system and how this influence can be reduced [1]. The process transfer function is given by

$$G(s) = \frac{1}{(0.01s + 1)(0.04s + 1)(0.2s + 1)(s + 1)}$$
Figure 11 shows a simulation for two PID controllers where the measurement noise has been set to 0.02, and the set-point and load disturbance amplitudes to zero. A PID controller with $K = 1$, $T_i = 1$, $T_d = 0.9$, $b = 0.5$, and $\kappa = 10$ is represented in red color. As seen, large control actions are obtained for this controller. This performance can be improved increasing the filtering effect [1]. So, the same controller is used but the $\kappa$ parameter is reduced to 1.5, showing the results in blue color. From the figure it can be observed how the control action variations are considerably reduced, remaining the system output practically unalterable.

The previous results can be corroborated from the gang of six transfer functions [1]. The transfer function which relates the control signal with the measurement noise is given by

$$G_{un} = \frac{C}{1 + PC} = CS = \frac{T}{P}$$

where, as measurement noise typically has high frequencies, this transfer function can be approximated by $G_{un} = C$. So, for a classical PID controller (1), at high frequencies $|G_{un}|$ becomes infinite due to the derivative term, which clearly indicates the necessity of filtering the derivative term. This fact can be tested for the previous time domain example, using the frequency domain mode in PID Basics. A simple index of the effect of measurement noise is the largest gain of the transfer function $G_{un}$ [1]

$$M_{un} = \max_\omega |G_{un}(i\omega)|$$

Figure 12 shows the frequency responses of $|G_{un}|$ for the previous two controllers with $\kappa = 10$ and $\kappa = 1.5$. The magnitude is considerably reduced for $\kappa = 1.5$ being the maximum value $\approx 3$, whereas for $\kappa = 10$ the maximum magnitude is $\approx 11$. 

18
Figure 12. Example of measurement noise response. Frequency domain interpretation for $\kappa = 10$ (left) and $\kappa = 1.5$ (right) using the transfer function $G_{in} = C/(1 + PC)$.

**PID LOOP SHAPING**

There are many interesting issues that have to be dealt with when developing IT for control which are related to the particular graphics representations used. It is straightforward to see the effects of parameters on the graphics but not so obvious how the graphical objects should be manipulated. There are natural ways to modify pole-zero plots, for example by adding poles and zeros and by dragging them. Bode plots can be manipulated by dragging the intersections of the asymptotes. Nevertheless, it is less obvious how a Nyquist plot should be changed. The tool presented in this section, called **PID Loop Shaping**, shows the Nyquist plots of the process transfer function $P(s)$ and the loop transfer functions $L(s) = P(s)C(s)$ (see Figure 13).

*Description of the Interactive Tool*

This section describes briefly the different elements of **PID Loop Shaping** and some easy theoretical aspects.
Figure 13. The user interface of the module *PID Loop Shaping*, showing both *Free* and *Constrained* PID tuning.

**Process:** This description of *PID Loop Shaping* is very similar to the *PID Basics* one. However, the process transfer function is presented as a pole-zero interactive graphic in the *s*-plane instead of a symbolic representation. The process transfer function can be modified depending on the option selected from the *Settings* menu. Several examples of transfer function are available, and its parameters can be modified using sliders as in *PID Basics*. However, a free transfer function can be selected (Interactive TF menu option), where poles and zeros can be defined graphically as Figure 13 shows.
**Controller:** This zone of the tool shows the different parameters and capabilities of **PID Loop Shaping** in order to perform loop shaping. Loop shaping is a design method where a controller is chosen such that the loop transfer function reaches the desired shape. The key idea is that the action of the controller can be interpreted as mapping the process Nyquist plot to the Nyquist plot of the loop transfer function. This mapping is performed to an specific design frequency \( \omega \). This frequency determines the desired point to move on the process transfer function, called *design point*. Such point is shown by a green circle on the L-plane graphic. The corresponding point at this frequency on the loop transfer function is called *target point*.

The controller representation used in **PID Loop Shaping** is given by the following parametrization [1]

\[
C(s) = k + \frac{k_i}{s} + k_d s
\]

The loop transfer function is thus

\[
L(s) = kP(s) + \left( \frac{k_i}{s} + k_d s \right) P(s)
\]

The point on the Nyquist curve of the loop transfer function corresponding to the frequency \( \omega \) is thus given by

\[
L(i\omega) = kP(i\omega) + i \left( -\frac{k_i}{\omega} + k_d \omega \right) P(i\omega)
\]

**PID Loop Shaping** provides three ways to tune the parameters in order to move the
process transfer function from the design point to the target point. These ways are shown at the section Tuning being called Free, Constrained PI, and Constrained PID. The first one allows performing the loop shaping dragging on the control parameters, and using the other two ones, the controller parameters are calculated based on some constrains on the target point. That is, the focus can be set on how the loop transfer function change when controller parameters are modified, or conversely, what parameters are required to obtain a given shape of the loop transfer function. For PI and PD control the mapping can be uniquely represented by mapping only one point \((x + yi)\). For PID control it is also possible to have an arbitrary slope of the loop transfer function at the target point \((x + yi, \text{ and } \theta)\). If the Free tuning option has been selected, some sliders appear in order to modify the controller gains \(k, k_i, \text{ and } k_d\) as shown in Figure 13, where the controller type can be chosen. The controller gains can also be changed by dragging arrows as illustrated in the same figure. From equation (2), the proportional gain changes \(L(i\omega)\) in the direction of \(P(i\omega)\), integral gain \(k_i\) changes it in the direction of \(-iP(i\omega)\), and derivative gain \(k_d\) changes it in the direction of \(iP(i\omega)\).

For Constrained PI and Constrained PID tuning options, the target point can be constrained to move on the unit circle, the sensitivity circles, or to the real axis. In this way it is easy to make loop shaping with specifications on gain and phase margins or on the sensitivities. In the case of Constrained PI it is necessary to find controller gains providing the desired target point. So, dividing equation (2) by \(P(i\omega)\) and taking real and imaginary parts [1]
\[ k = \Re \left( \frac{L(i\omega)}{P(i\omega)} \right) \]

\[-\frac{k_i}{\omega} + k_d \omega = \Im \left( \frac{L(i\omega)}{P(i\omega)} \right) = A(\omega) \tag{3}\]

Equation (3) gives directly the parameters of a PI controller with \( k_d = 0 \). An additional condition is required for Constrained PID tuning option. So, it is observed that

\[
L'(s) = C'(s)P(s) + C(s)P'(s) = C'(s)P(s) + \frac{L(s)P'(s)}{P(s)}
\]

\[
= \left( -\frac{k_i}{s^2} + k_d \right) P(s) + \frac{L(s)P'(s)}{P(s)}
\]

The slope of the Nyquist curve is then given by

\[
\frac{dL(i\omega)}{d\omega} = iL'(i\omega) = i\left( \frac{k_i}{\omega^2} + k_d \right) P(i\omega) + iC(i\omega)P'(i\omega)
\]

This complex number has the argument \( \theta \) if

\[
\Im \left( iL'(i\omega)e^{-i\theta} \right) = 0,
\]

which implies that

\[
\frac{k_i}{\omega^2} + k_d = \frac{\Re \left( L(i\omega)P'(i\omega)e^{-i\theta} \right)}{\Re \left( P(i\omega)e^{-i\theta} \right)} = B(\omega) \tag{4}\]

Combining equation (4) with equation (3) gives the controller parameters

\[
k_i = -\omega A(\omega) + \omega^2 B(\omega) \tag{5}\]

\[
k_d = \frac{A(\omega)}{\omega} + B(\omega)
\]
where $A(\omega)$ and $B(\omega)$ are given by equations (3) and (4).

The frequency design $\omega$, which determines the design point, can be chosen using the slider \texttt{wdesign} or graphically by dragging on the green circle on the process Nyquist curve (black curve). The target point on the Nyquist plot and its slope can be dragged graphically. The slope can also be changed using the slider \texttt{slope}. Furthermore, it is possible to constrain the target using the Constraints radio buttons. The target point can be constrained to the unit circle (\texttt{Pm}), the negative real axis (\texttt{Gm}), circles representing constant sensitivity (\texttt{Ms}), constant complementary sensitivity (\texttt{Mt}), or constant sensitivity combinations (\texttt{M}). When sensitivity constraints are active, the associated circles are drawn on the L-plane plot and some sliders appear in order to allow modifying their values. The circles are defined as follows [1]:

<table>
<thead>
<tr>
<th>Contour</th>
<th>Center</th>
<th>Radius</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_s$-circle</td>
<td>$-1$</td>
<td>$1/M_s$</td>
</tr>
<tr>
<td>$M_t$-circle</td>
<td>$-\frac{M_t^2}{M_t^2 - 1}$</td>
<td>$\frac{M_t}{M_t^2 - 1}$</td>
</tr>
<tr>
<td>$M$-circle</td>
<td>$-C_{st}$</td>
<td>$R_{st}$</td>
</tr>
</tbody>
</table>

where

\[
x_1 = \max\left(\frac{M_s + 1}{M_s}, \frac{M_t}{M_t - 1}\right), \quad x_2 = \max\left(\frac{M_s - 1}{M_s}, \frac{M_t}{M_t + 1}\right)\]

\[
C_{st} = \frac{x_1 + x_2}{2}, \quad R_{st} = \frac{x_1 - x_2}{2}
\]

Figure 13 illustrates designs for two PID controllers and a given sensitivity. The target
point is moved to the sensitivity circle and the slope is adjusted so that the Nyquist curve is outside of the sensitivity circle. The red design shows a PID controller using Free tuning, and the blue one a Constrained PID.

![Diagram](image)

Figure 14. Free (left) and constrained (right) tuning views for *PID loop shaping*.

**Robustness and performance parameters:** This zone is located below the controller parameters (see Figure 13), showing parameters which characterize robustness and performance in the same way that in *PID Basics*. The measures are maximum sensitivity (\(M_s\)), the sensitivity crossover frequency (\(W_s\)), maximum complementary sensitivity (\(M_t\)), the complementary sensitivity crossover frequency (\(W_t\)), gain margin (\(G_m\)), gain crossover frequency (\(W_{gc}\)), phase margin (\(P_m\)), and phase crossover frequency (\(W_{pc}\)).

**L-plane Graphic:** It represents the right side of *PID loop shaping*, as can be seen in Figure 13. This graphic contains the Nyquist plots of the process transfer function \(P(s)\) (in
black) and the loop transfer functions \( L(s) = P(s)C(s) \) (in red). Three different views can be shown depending on the tuning options. Figure 14 shows two views, one for free tuning and another one for constrained PID tuning. A third view is shown in Figure 13 where two designs are shown simultaneously. The design and target points can interactively be modified on this graphic. The design point is shown in green color on the Nyquist curve of the process. The target point is represented in light green color (in the case of free tuning), or in black color (for constrained tuning) as in Figure 13. The slope of the target point can also be changed interactively.

For free tuning, the controller gains are shown as arrows on the Nyquist plane. The controller gains can interactively be modified by dragging on the ends of the arrows. Examples of these arrows are shown in Figures 13 and 14. The scale of the graphic can be changed using the red triangle located at the bottom of the vertical axis.

As commented above, it is possible to impose constraints on the target point. The graphical representation of the target point is modified depending on the constraint selected, restricting its value based on its meaning. Therefore, different target point locations are given based on the active constraint as Figure 15 shows.

On the top of L-plane graphic the options save and delete can be found. These options have the same meaning that in PID Basics, being possible to save designs in order to perform comparisons. Once the save option is active, two pictures appear showing in red color the current design and in blue the second one (see Figure 13). Then, the modifications on the controller parameters affect to the current (active) design. The current design can be changed using the options Design 1 and Design 2 shown on the top of L-plane graphic. Thus, once the current
design is chosen, the associated curve is changed to red color and the controller zone is modified based on that design. The value of the controller gains for each design can be shown locating the mouse on the curves.

**Settings menu:** The Settings menu is available on the top menu of *PID Loop Shaping* and divided into four groups, following the same structure that in *PID Basics*. From the first
entry, Process transfer function, several processes can be selected, being also available two options to include free transfer functions. One of them, called String TF..., allows including a transfer function in a symbolic way. For example, \( P(s) = \frac{1}{\cosh \sqrt{s}} \) can be represented as \( P = '1/cosh(sqrt(s))' \). The other option is active to define the process transfer function using the interactive pole-zero representation at the Process zone. Results can be stored and recalled using the Load/Save menu. From this menu, data can be saved and recalled using the options Save design and Load design, respectively. The option Save report can be used to save all essential data in ascii format, being this possibility useful for documenting results. Specific values for control parameters can be entered by Parameters menu option. As in PID Basics, the last menu option (Examples Advanced PID Book) allows loading examples from the book [1].

Illustrative Examples

Some of the capabilities in PID Loop Shaping are shown by means of some examples.

Effect of controller parameters. Free tuning.: As commented above, understanding how the Nyquist plot of the compensated system changes based on the controller parameters is sometimes complicated. This example has the purpose of showing basic exercises about this issue.

Consider the same process that used in PID Basics to study load disturbances, where the transfer function is given by \( P(s) = \frac{1}{(s+1)^4} \). If a P-controller is used, the proportional gain changes the loop transfer function \( L(i\omega) = kP(i\omega) \) in the direction of \( P(i\omega) \). Figure 16(a)
Figure 16. Nyquist plot modifications depending on the controller type.

shows the effect of modifying \( L(i\omega) \) using two proportional controllers. The blue curve is for \( k = 2 \) and the red one for \( k = 2.6 \). So, it can be seen how the proportional gain modifies the Nyquist plot of the process (black curve) at frequency \( \omega \) (green circle on the black curve) in the direction of \( P(i\omega) \). Figure 16(b) shows the same study for an I-controller with \( k_i = 1 \) (red curve) and \( k_i = 0.6 \) (blue curve). It is observed how the integral gain \( k_i \) changes \( L(i\omega) \) in the direction \(-iP(i\omega)\) (the derivative gain has the same effect but in the direction \( iP(i\omega) \)). If PI or PD controllers are used, the compensated point at frequency \( \omega \) is calculated as the sum of two vectors, the proportional vector, and the integral or derivative one. Examples of this capability

29
are shown in Figures 16(c) and (d), where the process is controlled by a PI controller \((k = 2.3\) and \(k_i = 0.7)\) and PD controller \((k = 2.1\) and \(k_d = 3.35)\), respectively.

![Nyquist plot and Controller and robustness parameters](image)

(a) Nyquist plot  (b) Controller and robustness parameters

Figure 17. Proportional gain to reach the critical point \(-1 + 0j\).

Some easy exercises can be performed in order to gain skills on the Nyquist plane. For example, using the previous process, it can be of interest to obtain the gain for a proportional controller where the closed-loop system changes from stable to unstable. Before playing on PID Loop Shaping, the result can be calculated analytically as follows

\[
\angle L(i\omega) = \angle C(i\omega)P(i\omega) = -180 \Rightarrow \angle k \frac{1}{(j\omega + 1)^4} = -180 \Rightarrow \omega = 1
\]

\[
|L(i\omega)| = |C(i\omega)P(i\omega)| = | -1 + 0j| \Rightarrow |k \frac{1}{(j\omega + 1)^4}| = -1 \Rightarrow k = 4
\]

So, PID Loop Shaping can be used to interactively verify the result, as Figure 17 shows. This kind of exercises challenges the students and encourage them to make observations and relate theory with pictures in order to develop a broader and deeper understanding.
Figure 18. Example of loop shaping with $M_s < 1.5$.

On the other hand, free interactive designs can also be performed to compare the results with consolidated design methods. For example, *PID Loop Shaping* can be used to interactively design a PID controller for the previous process where the maximum sensitivity value is required to be less than 1.5 ($M_s \leq 1.5$). After playing with the IT, a PID controller which fulfils this constraint is obtained where $k = 0.92$, $T_i = 1.8$, $k_i = 0.5$, $T_d = 1.03$ and $k_d = 0.95$. Then, the AMIGO-frequency method can be used to perform the same design and to compare the results. The controller obtained is given by $k = 1.2$, $T_i = 2.48$, $k_i = 0.48$, $T_d = 0.93$ and $k_d = 1.12$. Figure 18 shows the Nyquist plots and time responses (using *PID Basics*) for both designs, in blue color for the free PID controller and in red for the AMIGO method. The obtained $M_s$ values are 1.49 for free PID and 1.46 for AMIGO method. The results are very similar, but the smaller $M_s$ value of AMIGO gives better robustness properties and load disturbances rejection.
Figure 19. Example of constrained design. Target point \(-0.5 - 0.5j\).

Effect of target point. Constrained designs.: The target point on the Nyquist plot can be reached using a free constraint design. Thus, the controller gains are interactively adjusted in order to perform this task as shown in the previous example. Nevertheless, another way is to use the equations (3), (4) and (5), where the controller gains are calculated once the target point is defined. As discussed above, the target point can freely be fixed, or constrained in different ways: any point \(x + yj\), or constrained to a specific value for phase margin, gain margin, or maximum values of the sensitivity functions. Figure 19 shows an example where the target point has been set to the point \(-0.5 - 0.5j\). Two constrained designs are shown for a design frequency \(\omega = 0.6\). The red curve represents a compensated system by a constrained PID with \(k = 1.32\), \(k_i = 1.02\) and \(k_d = 2.15\), while the blue one represents a constrained PI with \(k = 1.32\) and \(k_i = 0.15\). Both controllers reach the target point, but better results are obtained for the PID controller due to the
slope, allowable third degree of freedom (5), where for this example the slope \( \theta \) takes the value 22. PID controller provides better robustness properties getting \( M_s = 1.45, k_i = 1.02, G_m = 5.32 \), and \( P_m = 40.15 \), versus PI controller with \( M_s = 1.83, k_i = 0.15, G_m = 2.69 \), and \( P_m = 75.77 \).

Similar examples to restrict the target point for phase margin, gain margin, or maximum values of the sensitivity functions can be performed, as represented in Figure 15. Figure 20(a) shows an example where a combined sensitivity constraint is required for \( M_s \leq 2 \) and \( M_t \leq 2 \). This constraint is fulfilled in two different ways using a constrained PID (in red) and a constrained PI (in blue). Another example combining sensitivity function and gain margin constraints is shown in Figure 20(b), with the specification that the gain margin is equal to 3 and \( M_t \leq 2 \), maximizing the integral gain \( k_i \). So, the constraint gain margin is chosen and the target point is located in
such a way that $G_m = 3$. Then, a constrained PID controller is selected, being the design point and the slope modified until $M_s \leq 2$ and the integral gain is maximized. The controller obtained is given by $k = 1.38$, $k_i = 0.52$, and $k_d = 0.54$ for $\omega = 1.02$ and slope $= 32$.

![Nyquist plot](image1.png) ![Time domain responses](image2.png)

(a) Nyquist plot (b) Time domain responses

Figure 21. Derivative cliff example

The derivative cliff: This example is available at the Settings menu of PID Loop Shaping [1]. The process transfer function is the same as in previous examples ($P(s) = 1/(s + 1)^4$). It is desired to maximize integral gain $k_i$ subject to the robustness constraint $M_s \leq 1.4$. The obtained controller provides the parameters $k = 0.925$, $k_i = 0.9$, and $k_d = 2.86$ where the Nyquist plot of the loop transfer function is shown in red in Figure 21(a). It can be observed that the Nyquist curve has a loop. This phenomenon is called derivative cliff (a deeper explanation can be found in [1]) and it is due to the fact that the obtained controller has excessive phase lead, which is
obtained by having a PID controller with complex poles \((T_i < 4T_d, \text{ in this example } T_i = 0.33T_d)\). Figure 21\((b)\) shows, in red color, the time response of this controller obtaining oscillatory outputs. For comparison, the results for a controller with \(T_i = 4T_d\) are shown in blue color in Figures 21\((a)\) and \((b)\) with the controller parameters \(k = 1.1, k_i = 0.36\) and \(k_d = 0.9\). The responses for this controller are better, even under larger overshoot in response to load disturbance.

**Delayed system:** Dead-times appear in many industrial processes, usually associated with mass or energy transport, or due to the accumulation of a great number of low-order systems. Dead-times produce an increase in the system phase lag, therefore decreasing the phase and gain margins and limiting the response speed of the system (system bandwidth) [8], [10].

An example is described in what follows in order to see the influence of the time delay. The process transfer function is given by

\[
P(s) = P_n(s)e^{-t_d s} = \frac{e^{-10s}}{2s + 1}
\]

where \(P_n(s)\) represents the delay-free system \(P_n(s) = \frac{1}{2s+1}\).

The control for \(P_n(s)\) is performed using a PI controller with \(k = 2.5, k_i = 3.58, T_i = 0.7\), obtaining an infinite gain margin. Figure 22 shows the Nyquist plot and the time responses obtaining fast tracking results and very good load disturbances rejection (notice that \(k_i = 3.58\)). If the same PI controller is used to control the plant with delay \(P(s)\), the system becomes unstable as Figure 23 shows. Notice the circles on the Nyquist plot of the process (black). Dead-times augments the system phase as frequency increases in the form \(\varphi = -t_d \omega\) (being \(\varphi\) the phase lag.

35
Figure 22. Delayed system example. Control for free-delay system.

provided by the dead-time). Hence, the circles appear when large values of the system phase are represented in the Nyquist plot.

Therefore, in order to control the system, it is necessary to reduce its bandwidth, decreasing the values of the proportional and integral gains. Figure 24 shows the compensated system for the control parameters $k = 0.33$, $k_i = 0.07$, $T_i = 4.57$. The system becomes stable, but at cost of having very slow responses (see the time scale in Figure 22(b) and Figure 24(b)) and poor load disturbance rejection (notice that $k_i = 0.07$). Some other design techniques can be used to get better results [1], [6], [7].
Figure 23. Delayed system example. Unstable results.

Figure 24. Delayed system example. Stable system with bandwidth limitation.
**PID Windup**

Many aspects of PID control can be understood using linear models. There are, however, some important nonlinear effects that are very common even in simple loops with PID control. Integral windup can occur in loops where the process has saturations and the controller has integral action. When the process saturates, the feedback loop is broken and the integral may reach large values maintaining the control signal saturated for a long time, resulting in large overshoots and undesirable transients [1].

The purpose of this module is to facilitate the understanding of integral windup and a method for avoiding it (see [1]). There are many different ways to protect against windup. Tracking is a simple method illustrated in the block diagram in Figure 25. As well-known in the anti-windup schemes, the system remains free when the saturation is not active. However, when saturation occurs, the integral term in the controller is modified until the system is out of the saturation limit, where the modification of the integral element is not performed instantaneously but dynamically with a time constant $T_i$ [1].

The module shows process outputs and control signals for unlimited control signals, limited control signals without anti-windup, and limited control signals with anti-windup (see Figure 26). Process models and controller parameters can be selected in the same way as in the other modules. The saturation limits of the control signal can be determined either by entering the values or by dragging the lines in the saturation metaphor. The main aspects of the tool and some illustrative examples will be shown in the following paragraphs.
Description of the Interactive Tool

This section briefly describes the main aspects of *PID Windup*. The main screen of the tool is shown in Figure 26.

![PID controller with anti-windup](image)

Figure 25. PID controller with anti-windup.

**Process:** It corresponds to the same *Process* zone that in previous tools. It provides the same elements that in *PID Basics*, where a symbolic representation of the process transfer function is available, and the process parameters can be modified by interactive sliders or text edits (see Figure 26). The time delay is modified using a slider instead of a text edit as in *PID Basics* or *PID Loop Shaping*. Then, the time delay effect on the anti-windup mechanism can be easily analyzed.

**Controller:** This section contains the information about the controller parameters and actuator saturation. Three kind of controllers with integral action can be selected (I, PI, PID), where several sliders are available to modify the controller parameters including the tracking time constant $T_t$ used to reduce the integral effect. A saturation metaphor graphic is also available.
Figure 26. The user interface of the module *PID Windup*, showing windup phenomenon and anti-windup technique.

in this zone. This graphic allows determining the saturation limits dragging on the small red circle located on the upper saturation value (notice that a symmetric saturation has been used; this choice has been selected for pedagogical purposes).

*Graphics:* Time responses for process output, control signal, and integral action are available in three different graphics (*Process Output, Controller Output, Integral term*). In the same way that in *PID Basics*, there exist multiple interactive graphical elements to modify: set-point, load disturbance, measurement noise, and horizontal and vertical scales (see Figure

40
These three graphics can simultaneously represent the controlled system in linear mode, non-linear mode with windup phenomenon, and non-linear mode with anti-windup technique. Such representations can be configured using the checkboxes located on the top of Process Output graphic. There are three checkboxes called Linear, Windup, and Antiwindup to include the associated signal in all graphics, that contain a legend for the three signals, showing the linear mode in red, windup mode in blue, and antiwindup one in green color. An example can be seen in Figure 27(a) where the three modes are active.

A dotted pink vertical line which makes it easier to compare the time in the plots is also available as Figure 27(a) shows. On the other hand, the saturation limits can be modified using the dotted blue horizontal lines available in the Controller Output graphic (see Figure 27).

![Graphs showing different modes](image)

(a) Several modes simultaneously  
(b) Proportional bands

Figure 27. Signal representation in PID Windup module.

The notion of proportional band is useful to understand the windup effect and is included
in PID Windup. The proportional band is defined as the range of process outputs where the controller output is in the linear range \([y_{\min}, y_{\max}]\). For a PI controller, the proportional band is limited by

\[
y_{\min} = b y_{sp} + \frac{I - u_{\max}}{K}
\]

\[
y_{\max} = b y_{sp} + \frac{I - u_{\min}}{K}
\]

where \(I\) is the integral term of a PI controller.

The same expressions hold for PID control if the proportional band is defined as the band where the predicted output \(y_p = y + T_d \frac{dy}{dt}\) is in the proportional band \([y_{\min}, y_{\max}]\). The proportional band has the width \((u_{\max} - u_{\min})/K\) and is centered at \(b y_{sp} + I/K - (u_{\max} + u_{\min})/(2K)\).

On the top of figure Process Output there are two additional checkboxes called PB Windup and PB Antiwindup. The activation of these options shows the proportional bands for the windup and antwindup cases in the Process Output graphic. The proportional bands are shown as dotted green and blue signals respectively, as Figure 27(b) shows.

**Settings menu:** The Settings menu is defined following the same structure as in previous modules. The process transfer function can be chosen from the entry Process transfer functions, and numerical values of the parameters can be introduced using the selection Controller parameters. Essential data and results can be saved and recalled using the Load/Save menu options. The menu selection Simulation makes it possible to choose the simulation time, and to activate the Sweep option which can be used to show the results for several values of the
tracking time constant (as will be seen later in an example). Several examples from [1] can be loaded from the Examples entry.

**Illustrative Examples**

Some examples to explain integral windup are going to be described using PID Windup.

![Diagram of control system](image)

(a) Maximum at integral term  
(b) Maximum at process output

Figure 28. Example windup phenomenon. Integrator system.

**Understanding Windup Phenomenon:** The windup phenomenon can be studied using the first entry from the Examples option menu. This example has been extracted from [1] and uses a pure integrator process $P(s) = \frac{1}{s}$ controlled by a PI controller with $K = 1, T_i = 1.2, b = 1$ with the control signal limited to ±0.1. Figure 28(a) shows the time responses for this example. The control signal is saturated from the first time instant. The process output and the integral term are increasing while the control error is positive. Once the process output exceeds the set-point,
the control error turns to be negative, but the control signal remains saturated due to the large value of the integral term (windup phenomenon). Looking at the dotted pink vertical line in Figure 28(a), the integral term reaches its largest value at $t = 10$, when the error goes through zero. So, the control signal does not leave the saturation limit until the error has been negative for a sufficiently long time to let the integral part come down to a small level. An interesting aspect can be observed in Figure 28(b), as the control signal is working in linear mode when the process output reaches its maximum value.

The proportional band can be drawn in this example using the PB Windup checkbox (Figure 29(a)). It can be tested (using the vertical line) how the process output remains inside of the band while the control signal is working in linear mode, and outside in other case. The proportional band is too narrow because the system is working in non-linear mode during a long time. Hence, an interesting correspondence of the proportional band narrowness and the controller gain can be obtained. Large controller gains provide narrow proportional bands (more energetic control signals and therefore more saturation time), and small controller gains gives wider proportional bands. Figure 29(b) displays this comment, where the proportional controller gain has been reduced to 0.2, producing a wider proportional band.

**Anti-windup:** The previous example is useful to understand the anti-windup technique by visualization. The same controller parameters have been used ($K = 1$, $T_i = 1.2$, $b = 1$) and the tracking time constant has been set to $T_i = 1$. Figure 30(a) shows the results where outputs for controller with (in green) and without (in blue) anti-windup can be observed. The improvements achieved with anti-windup are notorious. Now, the system only remains in saturation for a short
Figure 29. Example windup phenomenon. Proportional band.

Figure 30. Example anti-windup technique. $T_i$ effect.
time period, being the output of the integral term considerably reduced. The proportional band for PI controller with anti-windup is shown in the same figure. It can be observed how a wider proportional band than for PI without anti-windup (Figure 30(a)) is obtained, remaining the process output inside of it most of the time. The effect of the tracking time constant is illustrated in Figure 30(b) for \( T_i = 0.1, 10, 50 \) (the Sweep menu option has been used). Very large values of \( T_i \) bring the result to the windup phenomenon producing large integral signals, while small values reset the integral term quickly getting better results. It may thus seem advantageous to have always very small values of \( T_i \). However, the next example will show some situations where this choice is not always advisable.

**The tracking time constant:** The tracking time constant is an important parameter because it determines the rate of resetting the integral term of the controller. It seems to be advantageous to have a small value for this constant. Nevertheless, measurement errors may accidentally reset the integral term if the tracking time constant is too small. The following example tries to explain this fact, where there is a measurement error in the form of a short pulse. The transfer function of the process is given by

\[
P(s) = \frac{1}{(0.5s + 1)^2}
\]

and the system is controlled using a PID with \( K = 3.5, T_i = 0.52, T_d = 0.14, \ K = 10, b = 1, \) and \( T_i = 1. \)

Figure 31(a) shows the control results. Notice the large transient after the pulse. The integral term is excessively reduced, even reaching negative values. Some simple rules that have been suggested for the tracking time, \( T_i = \sqrt{T_i T_d} \) and \( T_i = (T_i + T_d)/2 \) [1], can be used to tune \( T_i \)
and avoid these problems. Figure 31(b) shows an example with $T_i = (T_i + T_d)/2 = 0.33$ where the response has been considerably improved.

![Graphs showing response comparison](image)

(a) Reset by measurement noise  (b) Tuning using rules

Figure 31. Tracking time tuning.

**CONCLUSION**

In this work a set of Interactive Modules have been presented as support for teaching basic Automatic Control concepts. These tools have as main objective the addition of interactivity to the visual content of the book *Advances PID Control* [1]. The modules are focused on PID control studying feedback fundamentals from the point of view of the time and frequency domains, including the robustness problem, measurement noise filtering, load disturbance rejection, and windup phenomenon.

The role of Interactivity in Automatic Control education has been shown presenting the
powerful of this element in Teaching. In our personal experience, Interactivity is an excellent element as support to teaching and learning which allows to enhance the motivation and participation of the future engineers. Authors animate to the readers to test these new interactive features in control education and engineers’ training.

REFERENCES


48


Corresponding author: José Luis Guzmán Sánchez.

Address: Universidad de Almería, Dpto. de Lenguajes y Computación, Ctra. Sacramento s/n, 04120, Almería, Spain. Telephone number: +34 950015849. Fax number: +34 950015129. Email: joguzman@ual.es.